5-21-2007

Using SAWSDL for Semantic Service Interoperability

Kunal Verma
Wright State University - Main Campus

Amit P. Sheth
Wright State University - Main Campus, amit@sc.edu

Using SAWSDL for Semantic Service Interoperability

Kunal Verma

Accenture Technology Labs

k.verma@accenture.com

Amit Sheth,

Karthik Gomadam

Kno.e.sis Center, **Wright State University**

http://knoesis.wright.edu

Introduction

- Semantic Annotations for WSDL (SAWSDL) is W3C Candidate Recommendation
- It defines a mechanism to add semantic annotations to Web services
- It is based on W3C member submission WSDL-S
  - WSDL-S was proposed by University of Georgia METEOR-S Team* and IBM
- It is an important first step by W3C to add support for semantic modeling the Web service stack.

Now mainly at Kno.e.sis Center, Wright State University

2

Details about SAWSDL

- **Standard Activity**
  - W3C SAWSDL Working group
    - http://www.w3.org/2002/ws/sawsdl/
  - W3C WSDL-S member submission Web page
    - http://www.w3.org/Submission/WSDL-S/
- **Tools**
  - SAWSDL4J by Wright State University (Dayton,OH) and University of Georgia (UGA, Athens,GA)
  - Radiant: WSDL-S/SAWSDL Annotation Tool by University of Georgia
  - Semantic Tools for Web Services by IBM alphaWorks
  - WSMO Studio by DERI
- **Some Relevant Papers**
  - Kunal Verma, Amit P. Sheth, Semantically Annotating a Web Service, IEEE Internet Computing, March/April 2007, Volume 11(2), pp. 83-85.
  - K. Verma, *Configuration and Adaptation of Semantic Web Processes,* PhD thesis, Dept. of Computer Science, Univ. of Georgia, Aug. 2006
  - K. Sivashanmugam, Kunal Verma, Amit Sheth, John A. Miller, Adding Semantic to Web Service Standards, ICWS 2003

Part 1

BACKGROUND AND MOTIVATION

Outline

- Part 1 – Background and Motivation
  - Evolution of SOA
    - Value proposition
    - Web Services Stack
    - Web Service Description Language
  - Adding Semantics to SOA
    - Motivation
    - Value proposition
    - How will Semantics Change SOA
- Part 2: Deep Dive into SAWSDL
- Part 3: Using SAWSDL
- Part 4: SAWSDL Tools

Semantic Oriented Architecture (SOA)

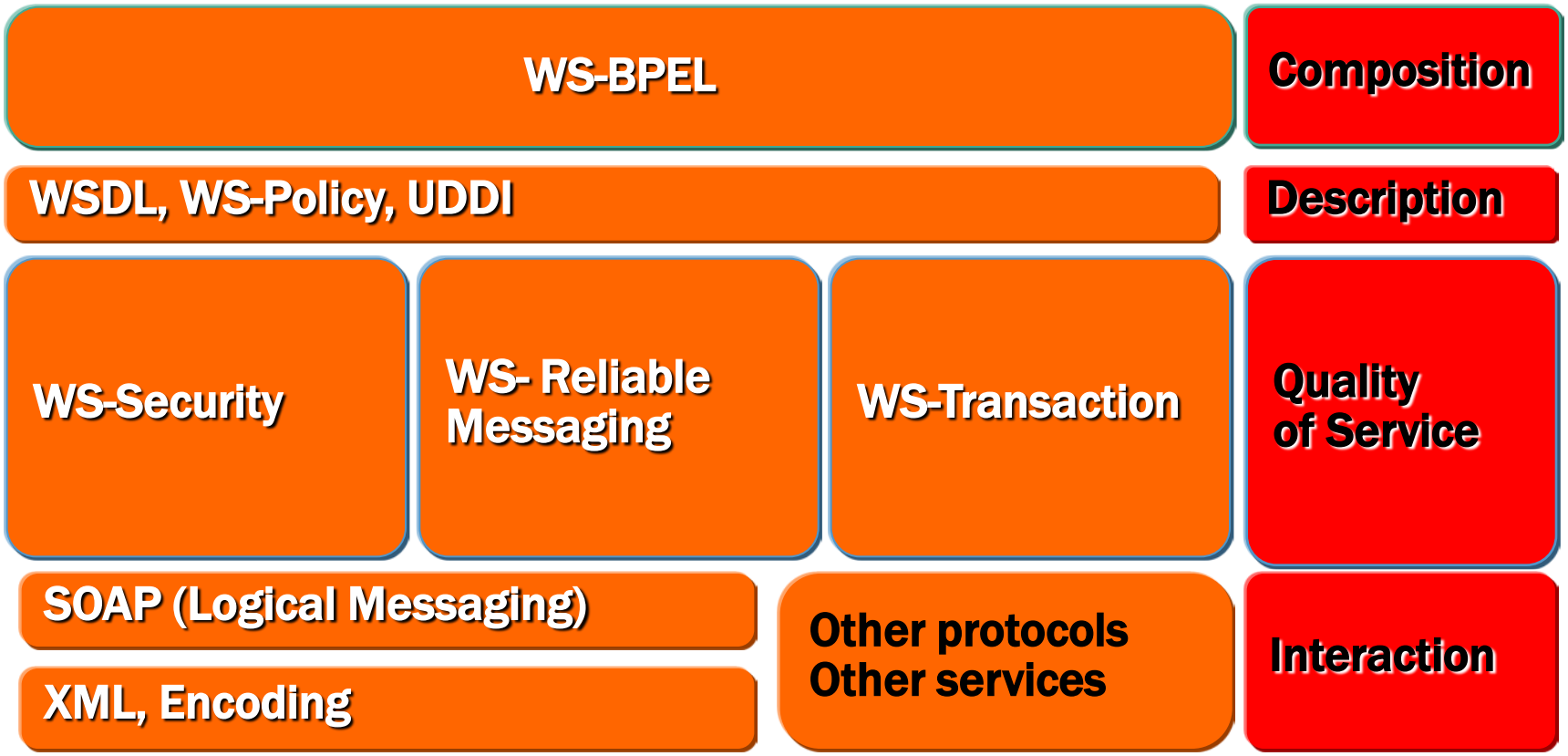
- SOA has transitioned from a vision to commonly implemented architecture

- Web services are a set of XML based standards
  - Front-runners to implement SOA based solutions
  - Widely supported by a number of vendors

Value Proposition of SOA and Web Services

- Interoperability
  - Ability to connect applications across platforms, languages, operating systems, geographic regions
- Re-use
  - Ability to re-use services across applications
- Insight
  - Direct visibility into business processes. Ability to link business goals to business processes
- Agility
  - Ability to react, ability to quickly create new processes

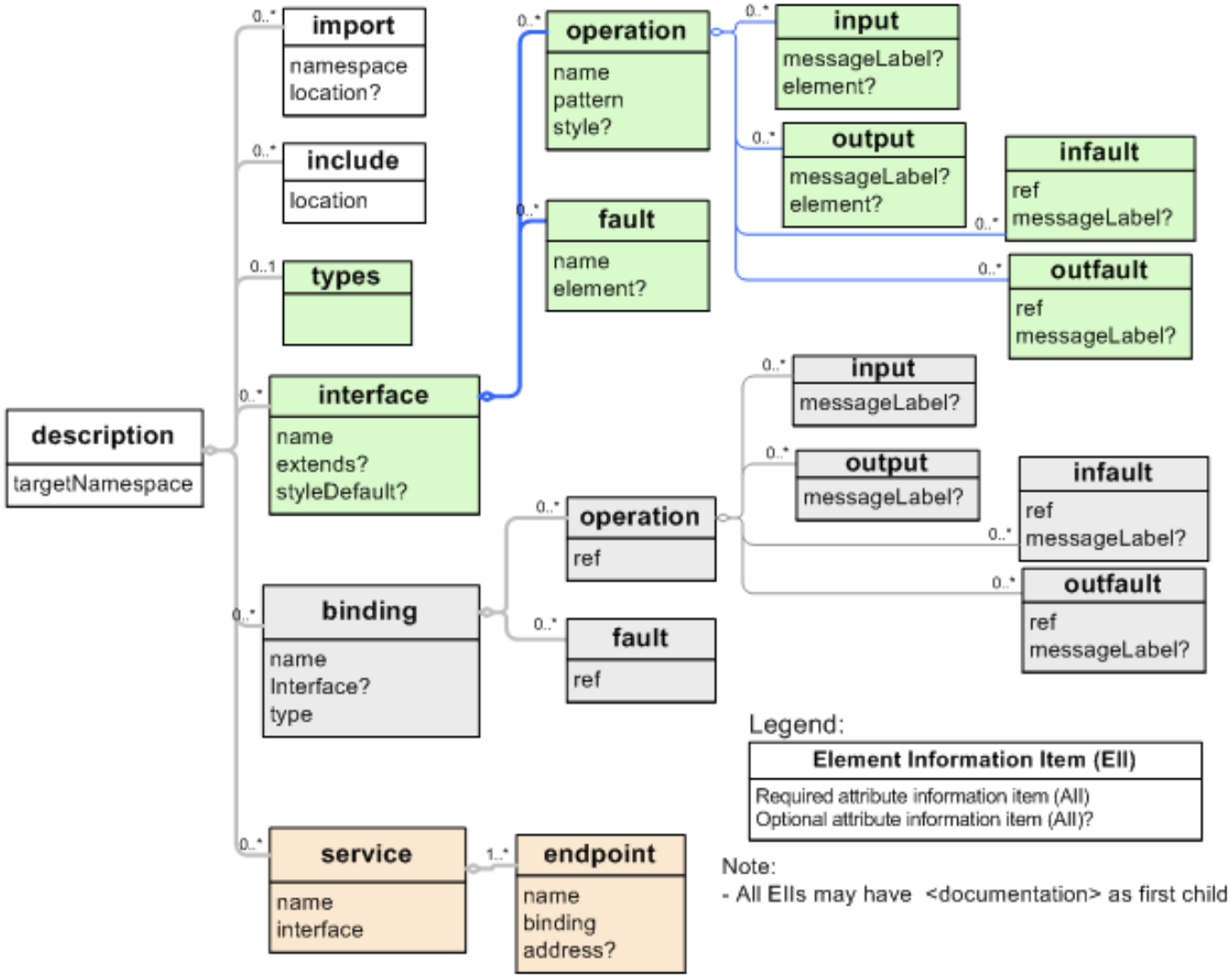
Web Service Standards



What is WSDL

- An extensible, platform independent XML language for “describing” services.
- Provides functional description of Web services:
  - IDL description
  - Access protocol and deployment details
  - All of the functional information needed to programmatically access a service, contained within a machine-readable format
- Does not include
  - QoS
  - Taxonomies
  - Business information
- WSDL is a component definition language for Web service component

WSDL 2.0 Component Model



Picture from: http://www.w3.org/TR/wsdl20-primer/

WSDL Example

```
<wsdl:description targetNamespace="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
   xmlns:wsdl="http://www.w3.org/ns/wsdl" xmlns:xs="http://www.w3.org/2001/XMLSchema" >
```

Namespaces

```
    <wsdl:types>
      <xs:element name= "processPurchaseOrderResponse" type="xs:string
        <xs:complexType>
          <xs:sequence>
             <xs:element name=“orderConfirmation" type="xs:integer" />
            <xs:element name=“deliveryDate" type=" xs:integer " />
          </xs:sequence>
        </xs:complexType>
     </xs:element>
    </wsdl:types>
```

Types

```
  <wsdl:interface name="PurchaseOrder“>
    <wsdl:operation name=“order” pattern=wsdl:in-out>
        <input messageLabel = ”processPurchaseOrderRequest"
           element="tns:processPurchaseOrderRequest"/>
        <output messageLabel ="processPurchaseOrderResponse"
           element="processPurchaseOrderResponse"/>
     </wsdl:operation>
     <wsdl:operation name=“cancel” pattern=wsdl:in-out>
        <input messageLabel = ”processCancelRequest"
          element="tns:processCancelRequest"/>
        <output messageLabel ="processCancelResponse"
          element="processCancelResponse"/>
     </wsdl:operation>
   </wsdl:interface>
</wsdl:description >
```

Operation

Operation

Interface

ADDING SEMANTICS TO SOA

Semantic Web Services

- Rich research history– too much to review here
- SWS related submissions to W3C
  - OWL-S: http://www.w3.org/Submission/OWL-S/
  - WSMO: http://www.w3.org/Submission/2005/06/
  - SWSF: http://www.w3.org/Submission/SWSF/
  - WSDL-S: http://www.w3.org/Submission/WSDL-S/
- W3C Workshop at Innsbruck, leading to community agreement to focus on limited scope and evolutionary approach championed by WSDL-S, leading to SAWSDL WG

Semantics to Web Services: The ingredients

- Conceptual Model/Ontology
  - An agreed upon model that captures the semantics of domain
  - Common Nomenclature
  - Domain Knowledge (facts)
- XML based service description
  - Standards and specifications like WSDL for web service description, WS-Agreement for capturing agreements etc.
- Annotate the service description

14

What does Semantics bring to the table?

- Better Reuse
  - Semantic descriptions of services to help find relevant services
- Better Interoperability
  - Beyond syntax to semantics, mapping of data exchanged between the services (very time consuming without semantics, just as XML in WSDL gives syntactic interoperability, SAWSDL gives semantic interoperability)
- Configuration/Composition
  - Enable dynamic binding of partners
- Some degree of automation across process lifecycle
  - Process Configuration (Discovery and Constraint analysis)
  - Process Execution (Addressing run time heterogeneities like data heterogeneities.)

15

SAWSDL

- Offer an evolutionary and compatible upgrade of existing Web services standards

- Externalize the semantic domain models
  - agnostic to ontology representation languages (although W3C recommended RDFS or OWL are likely to be often used)
  - reuse of existing domain models (in some domains, usable ontologies have been built, eg life science and health care)
  - allows annotation using multiple ontologies (same or different domain)

- Updating tools around WSDL is relatively easier

16

Guiding principles...

- Support semantic annotation of Web Services whose data types are described in XML schema

- Provide support for rich mapping mechanisms between Web Service schema types and ontologies

17

Why use SAWSDL

- Build on existing Web Services standards using only extensibility elements
- Mechanism independent of the semantic representation language (though OWL is supported well)
- SAWSDL provides an elegant solution
  - Help integration by providing mapping to agreed upon domain models (ontologies, standards like Rosetta Net, ebXML)
  - Better documentation by adding functional annotation
- Ease in tool upgrades
  - e.g. wsif / axis invocation

18

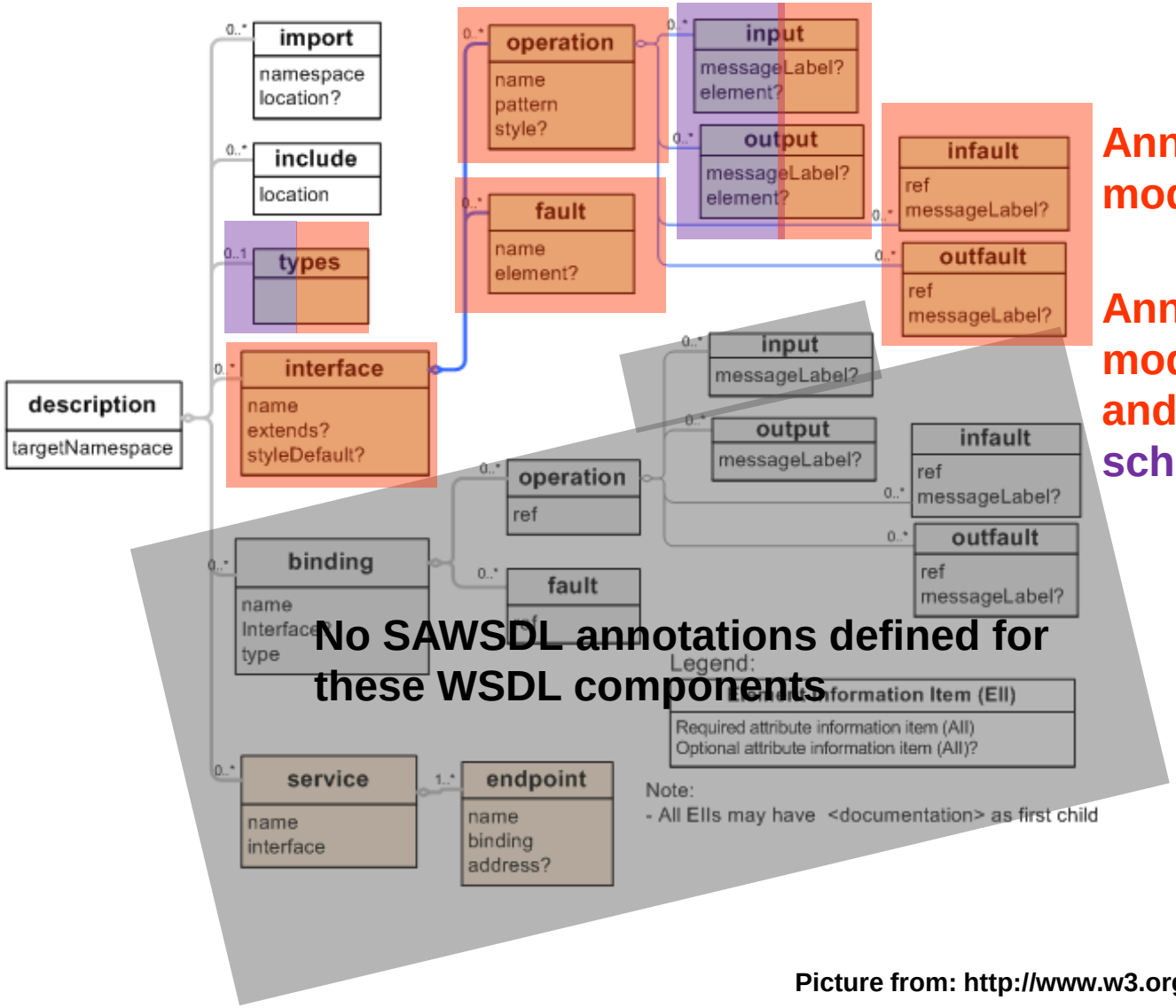
Part 2

DEEP DIVE INTO SAWSDL

Outline

- Part 1 – Background and Motivation
- Part 2: Deep Dive into SAWSDL
  - SAWSDL Scope
  - The extensibility attributes
  - Annotating operations
  - Annotating faults
  - Annotating types
  - Annotating interfaces
  - SAWSDL Example
- Part 3: Using SAWSDL
- Part 4: SAWSDL Tools

SAWSDL Scope

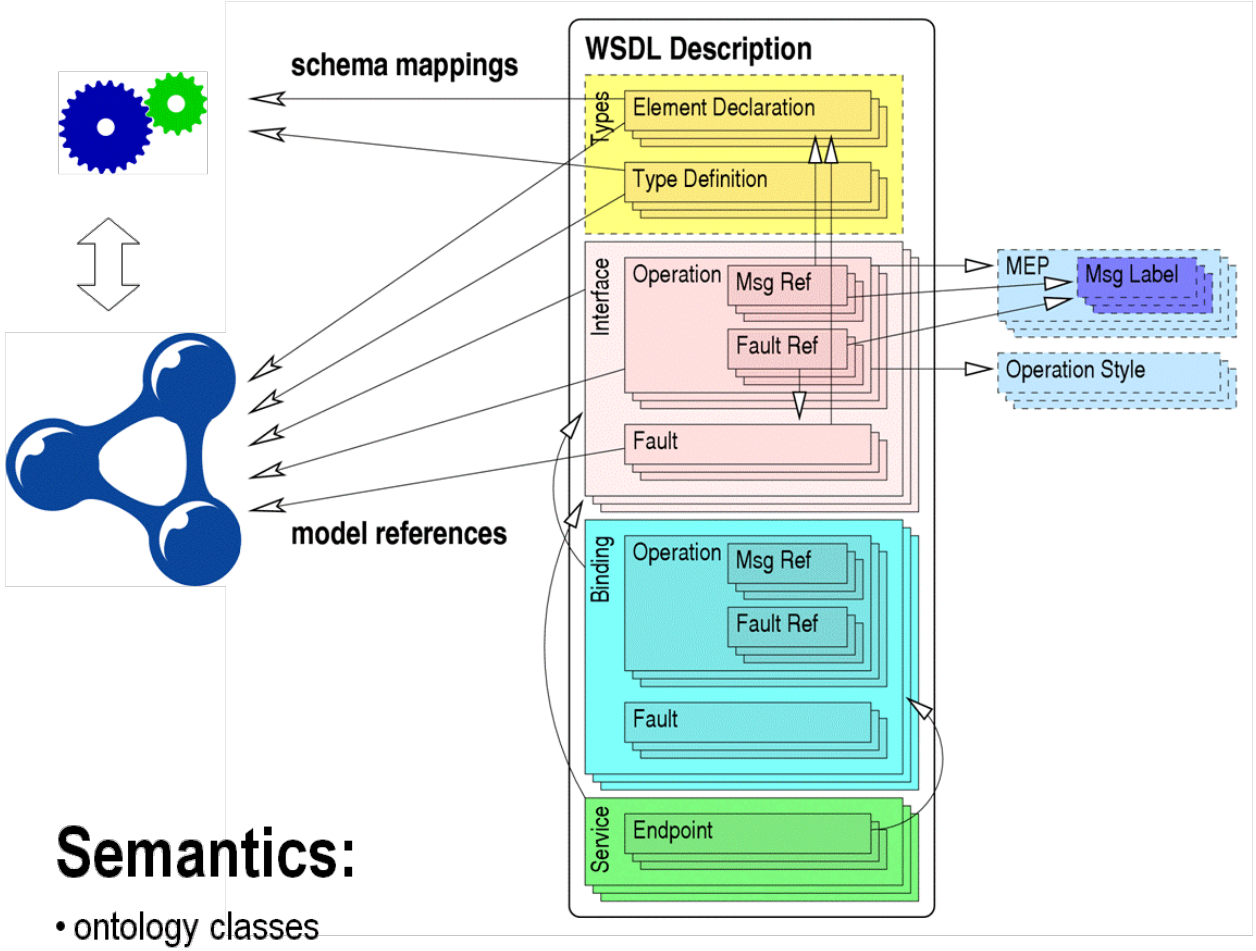


Annotated using modelReference

Annotated using modelReference and schemaMapping

Picture from: http://www.w3.org/TR/wsdl20-primer/

SAWSDL at a glance



Semantics:

- ontology classes
  - discovery, composition
  - filtering, ranking
- lifting/lowering mappings
  - mediation, invocation
- functionality categories
  - publishing, discovery, composition
- anything, really

22

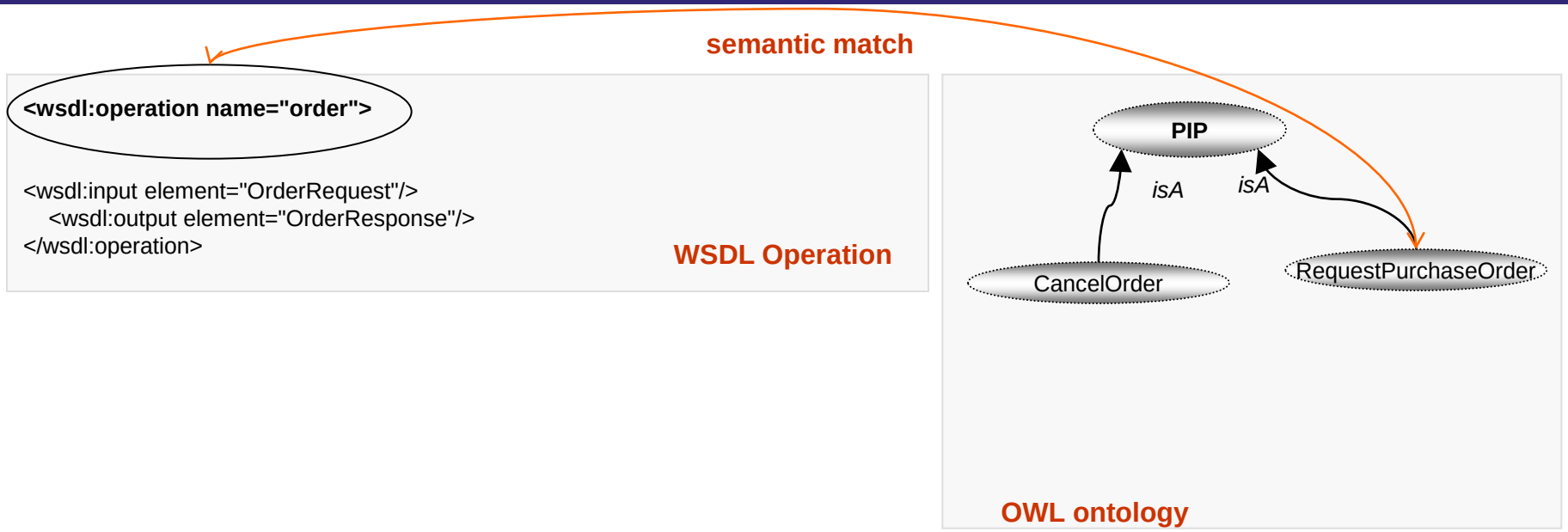
Ack: Jacek Kopecky

SAWSDL defines two extensibility attributes

- **modelReference:** This can be used to specify the association between a WSDL or XML Schema component and a concept in some semantic model.
  - It can be used to annotate the following:
    - WSDL components
      - Interfaces
      - Operations
      - faults
    - WSDL Type Definitions
      - XML Schema complex type definitions
      - Simple type definitions
      - element declarations
      - attribute declarations
- liftingSchemaMapping: This can be used to specify mappings between WSDL Type Definitions in XML and semantic data.
- loweringSchemaMapping: This can be used to specify mappings between semantic data and WSDL Type Definitions in XML.

23

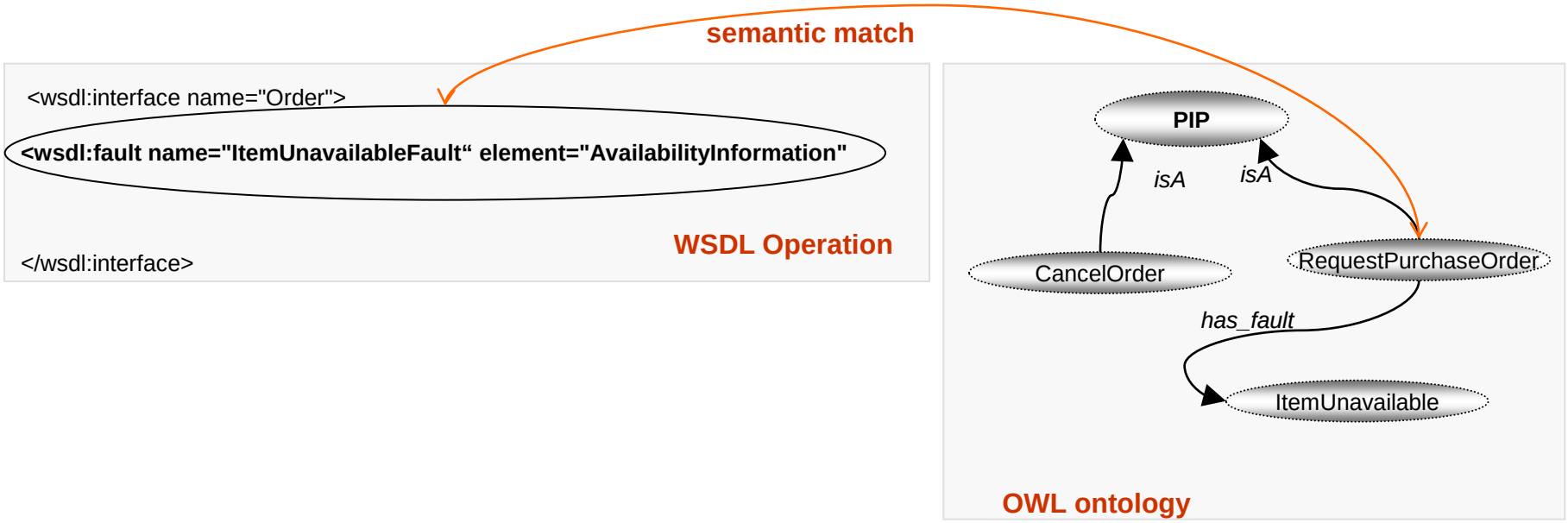
Using modelReference to annotate operations




```
<wsdl:operation name="order"
   sawsdl:modelReference="http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#RequestPurchaseOrder">
   <wsdl:input element="OrderRequest"/>
   <wsdl:output element="OrderResponse"/>
</wsdl:operation>
```

The annotation of the *operation* element carries a reference to a concept in a semantic model that provides a high level description of the operation, specifies its behavioral aspects or includes other semantic definitions.

Using modelReference to annotate faults



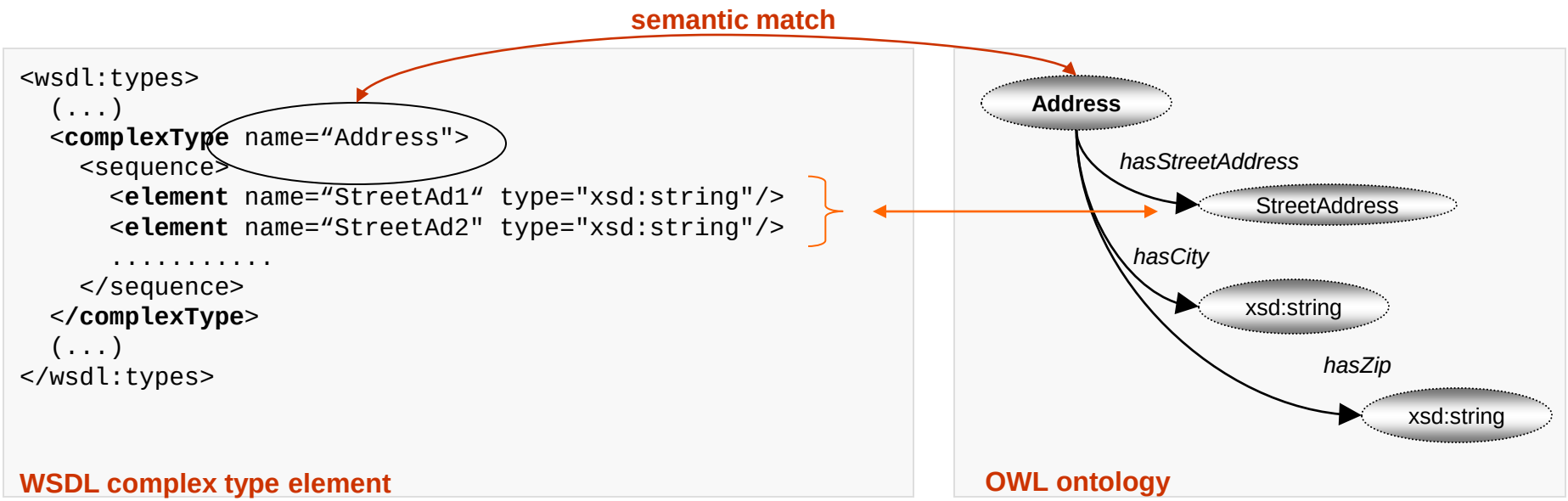

```
<wsdl:interface name="Order">
   <wsdl:fault name="ItemUnavailableFault" element="AvailabilityInformation"
      sawsdl:modelReference="http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#ItemUnavailable"/>
</wsdl:interface>
```

The annotation of the fault element carries a reference to a concept in a semantic model that provides a high level description of the fault and can include other semantic definitions

Annotating Types

- Following WSDL Type Definitions can be annotated using the modelReference, liftingSchemaMapping and loweringSchemaMapping extension attributes
  - XML Schema complex type definitions
    - Bottom-level annotation
    - Top level annotation
  - Simple type definitions
  - element declarations
  - attribute declarations

Annotating types



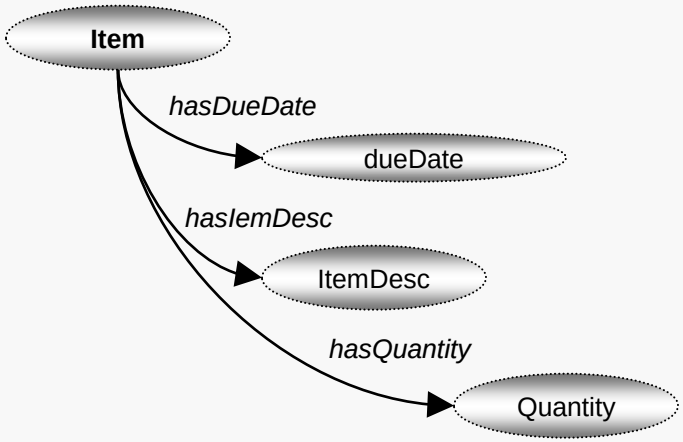
1. modelReference to establish a semantic association
2. liftingSchemaMapping and loweringSchemaMapping to provide mappings between XML and semantic model

27

Annotating complex types with modelReference – Bottom level annotation

```
<complexType name="POItem" >
 <all>
  <element name="dueDate" nillable="true" type="dateTime"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#DueDate”/>
  <element name="qty" type="float"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#Quantity”/>
  <element name="EANCode" nillable="true" type="string"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#ItemCode”/>
  <element name="itemDesc" nillable="true" type="string"
    sawsdl:modelReference=”
   http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#ItemDesc” />
  </all>
</complexType>
```

WSDL complex type element



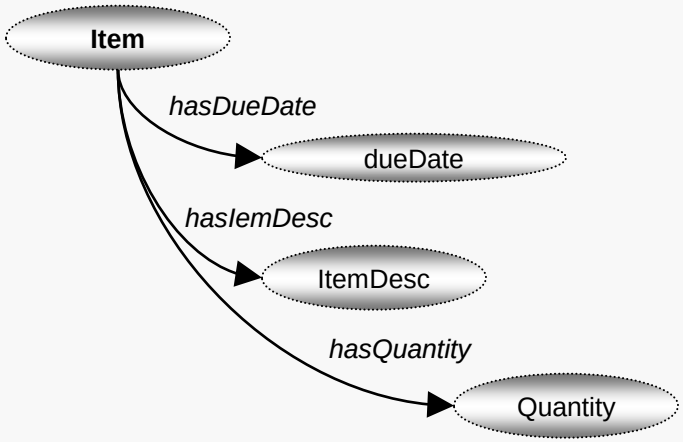
OWL ontology

28

Annotating complex types with modelReference – Top level annotation

```
<complexType name="POItem“
 sawsdl:modelReference=”
        http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#DueDate >
  <all>
     <element name="dueDate" nillable="true" type="dateTime" />
     <element name="qty" type="float"/>
     <element name="EANCode" nillable="true" type="string" />
     <element name="itemDesc" nillable="true" type="string" />
   </all>
</complexType>
```

WSDL complex type element



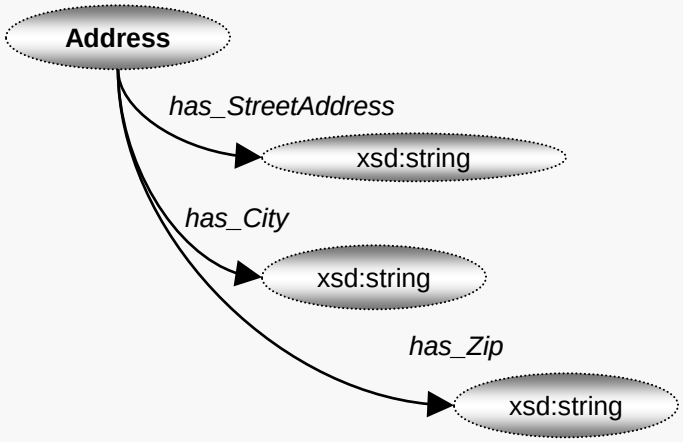
OWL ontology

29

Using schemaMapping with modelReference

```
<complexType name="POAddress"
  sawsdl:modelReference=
     "http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#Addresd"
  sawsdl:liftingSchemaMapping=
     http://www.w3.org/2002/ws/sawsdl/spec/mapping/POAdress2Ont.xslt
  sawsdl:loweringSchemaMapping=
     "http://www.w3.org/2002/ws/sawsdl/spec/mapping/Ont2POAddress.xslt>
   <all>
    <element name="streetAddress" type="string" />
    <element name="poBox" type="string" />
    <element name="city" type="string" />
    <element name="zipCode" type="string" />
    <element name="state" type="string" />
    <element name="country" type="string" />
    <element name="recipientInstName" type="string" />
   </all>
</complexType>
```

WSDL complex type element



OWL ontology

- Any mapping language can be used for liftingSchemaMapping
  - Recommended languages: XSLT, Xquery
- Any mapping language can be used for liftingSchemaMapping
  - Recommended languages: SPARQL to query ontology, followed by XSLT, Xquery

liftingSchemaMapping example using XSLT

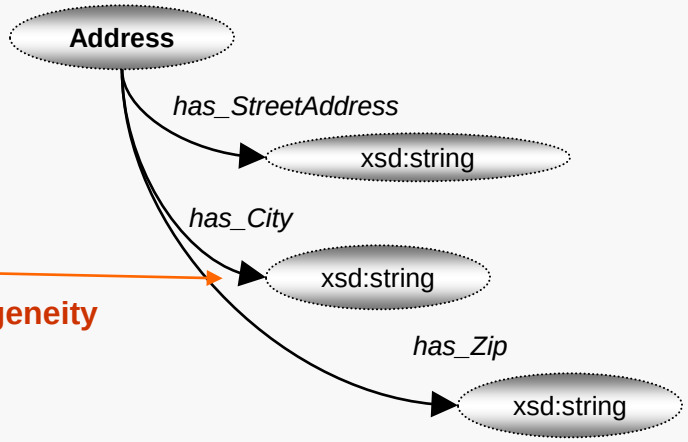
```
<xsl:transform version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
     ns#" xmlns:po="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
     xmlns:POOntology="http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#">
   <xsl:output method="xml" version="1.0" encoding="iso-8859-1" indent="yes" /> <xsl:template match="/">
   </POOntology:OrderConfirmation>
    <POOntology:Address rdf:ID="Address1">
     <POOntology:has_StreetAddress rdf:datatype="xs:string">
        <xsl:value-of-select="concat(POAddress/streetAddress)"/>
      </POOntology:has_StreetAddress >
      <POOntology:has_City rdf:datatype="xs:string">
        <xsl:value-of select="POAddress/city"/>
      </POOntology:has_City>
      <POOntology:has_Zip rdf:datatype="xs:string">
         <xsl:value-of select="POAddress/zip"/>
       </POOntology:has_State>
  </xsl:template>
</xsl:transform>
```

Using schemaMapping with modelReference (heterogeneity)

```
<complexType name="POAddress"
  sawsdl:modelReference=
     "http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#Addresd"
  sawsdl:liftingSchemaMapping=
     http://www.w3.org/2002/ws/sawsdl/spec/mapping/POAdress2Ont.xslt
  sawsdl:loweringSchemaMapping=
     "http://www.w3.org/2002/ws/sawsdl/spec/mapping/Ont2POAddress.xslt>
   <all>
    <element name="streetAddr1" type="string" />
    <element name="streetAddr2" type="string" />
    <element name="poBox" type="string" />
    <element name="city" type="string" />
    <element name="zipCode" type="string" />
    <element name="state" type="string" />
    <element name="country" type="string" />
    <element name="recipientInstName" type="string" />
   </all>
</complexType>
```

WSDL complex type element

Data level heterogeneity

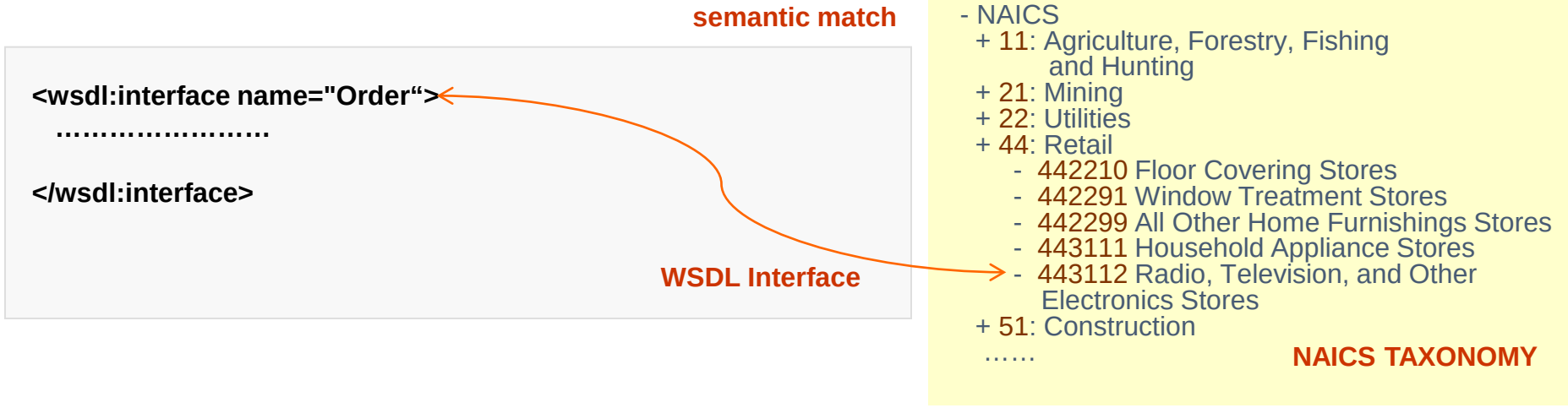


OWL ontology

liftingSchemaMapping example using XSLT (heterogeneity)

```
<xsl:transform version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:po="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
    xmlns:POOntology="http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#">
   <xsl:output method="xml" version="1.0" encoding="iso-8859-1" indent="yes" /> <xsl:template
    match="/">
   </POOntology:OrderConfirmation>
    <POOntology:Address rdf:ID="Address1">
      <POOntology:has_StreetAddress rdf:datatype="xs:string">
        <xsl:value-of-select="concat(POAddress/streetAddr1,POAddress/streetAddr2)"/>
      </POOntology:has_StreetAddress >
      <POOntology:has_City rdf:datatype="xs:string">
        <xsl:value-of select="POAddress/city"/>
      </POOntology:has_City>
      ……
  </xsl:template>
</xsl:transform>
```

Using modelReference to annotate interfaces




```
<wsdl:interface name="Order"
      sawsdl:modelReference="http://example.org/categorization/products/electronics">
    …
</wsdl:interface>
```

A *modelReference* on a WSDL *interface* element provides a reference to a concept or concepts in a semantic model that describe the Interface.

SAWSDL Example

```
<wsdl:description targetNamespace="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
   xmlns:wsdl="http://www.w3.org/ns/wsdl" xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:sawsdl="http://www.w3.org/ns/sawsdl">
   <wsdl:types>
     <xs:element name= "processPurchaseOrderResponse" type="xs:string
       sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#PurchaseOrderResponse”
       sawsdl:liftingSchemaMapping= http://www.w3.org/2002/ws/sawsdl/spec/mapping/POResponse2Ont.xslt
       sawsdl:loweringSchemaMapping= "http://www.w3.org/2002/ws/sawsdl/spec/mapping/Ont2Response.xslt>
        ……
    </xs:element>
  </wsdl:types>
  <interface name="PurchaseOrder"
  < sawsdl:modelReference="http://example.org/categorization/products/electronics />
   <operation name=“order” pattern=wsdl:in-out
        sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#RequestPurchaseOrder"" >
        <input messageLabel = ”processPurchaseOrderRequest"
           element="tns:processPurchaseOrderRequest"/>
        <output messageLabel ="processPurchaseOrderResponse"
           element="processPurchaseOrderResponse"/>
   </operation>
   <operation name=“cancel” pattern=wsdl:in-out
        sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#CancelOrder"" >
        <input messageLabel = ”processCancelRequest"
          element="tns:processCancelRequest"/>
        <output messageLabel ="processCancelResponse"
          element="processCancelResponse"/>
    </operation>
  </interface>
</wsdl:description >
```

Part 3

USING SAWSDL

Outline

- Part 1 – Background and Motivation
- Part 2: Deep Dive into SAWSDL
- Part 3: Using SAWSDL
  - Using SAWSDL with UDDI for Discovery
  - Using SAWSDL with WS-BPEL for dynamic binding
  - Using SAWSDL with Apache Axis for Data Mediation
- Part 4: SAWSDL Tools

Part 3.1

USING SAWSDL WITH UDDI FOR DISCOVERY

Using SAWSDL with UDDI

- The semantic annotations in SAWSDL represent the semantic signature of a services
  - Using annotations from semantic models can help with discover services with certain semantic signatures

- As part of METEOR-S project, we investigated publishing WSDL-S/SAWSDL files in UDDI Registries [1]

- Builds upon following previous discovery implementations
  - Extended matching presented in [2] to consider operations and service level metadata
  - Extends the approach presented “WSDL to UDDI Mapping” [3] to get SAWSDL to UDDI Mapping

[1] K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar and John Miller, METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services, JITM, Jan 2005

[2] M. Paolucci, T. Kawamura, T. Payne and K. Sycara, Semantic Matching of Web Services Capabilities, ISWC 2002.2

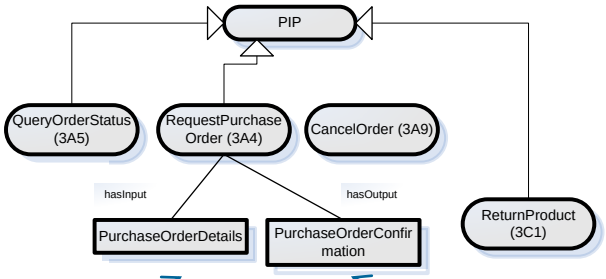
[3] Using WSDL in a UDDI Registry, Version 2.0.2 - Technical Note, http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v202-20040631.pdf

SAWSDL publication and discovery using UDDI

accenture

kno.e.sis

Semantic Model



1. SAWSDL file creating using annotations (modelReferences) pointing to semantic model

3. Service request created using terms from semantic model

SAWSDL File

Service Request (Semantic Template)

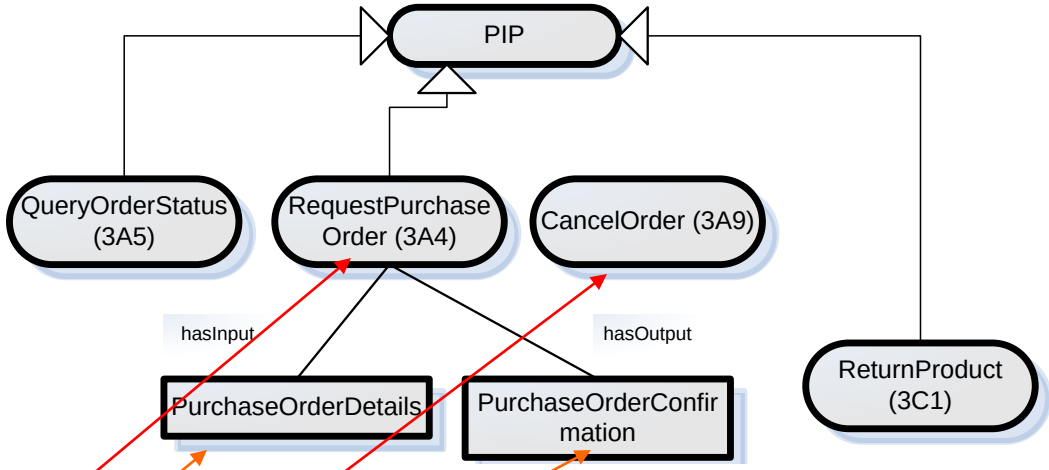
2. Service published in UDDI along with annotations

UDDI Registry

4. Discovery based on annotations

Using SAWSDL to represent semantic templates

- **Semantic Templates** capture the functionality of a Web service with the help of ontologies/other domain models
- Find a service that sells RAM in Athens, GA. It must allow the user to return and cancel, if needed
- Semantic templates can also have non-functional (QoS) requirements such as response time, security, etc. using WS-Policy



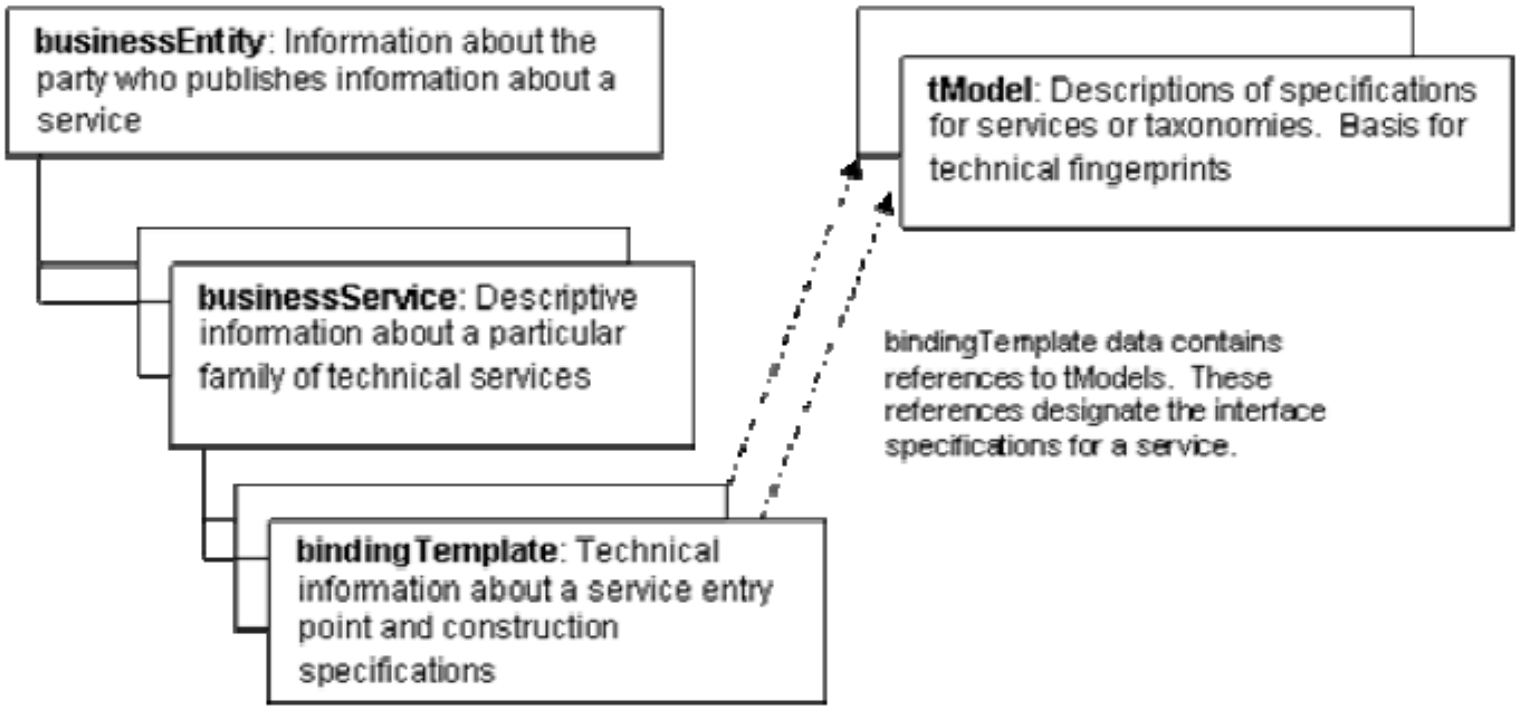
Part of Rosetta Net Ontology

SEMANTIC TEMPLATE

Service Level Metadata (SLM) IndustryCategory = NAICS:Electronics ProductCategory = DUNS:RAM Location = Athens, GA
Operation 1 Operation-modelReference = Rosetta#RequestPurchaseOrder Input-modelReference = Rosetta#PurchaseOrderRequest Output-modelReference = Rosetta#PurchaseConfirmation
Operation 2 Operation-modelReference = Rosetta#CancelOrder ……

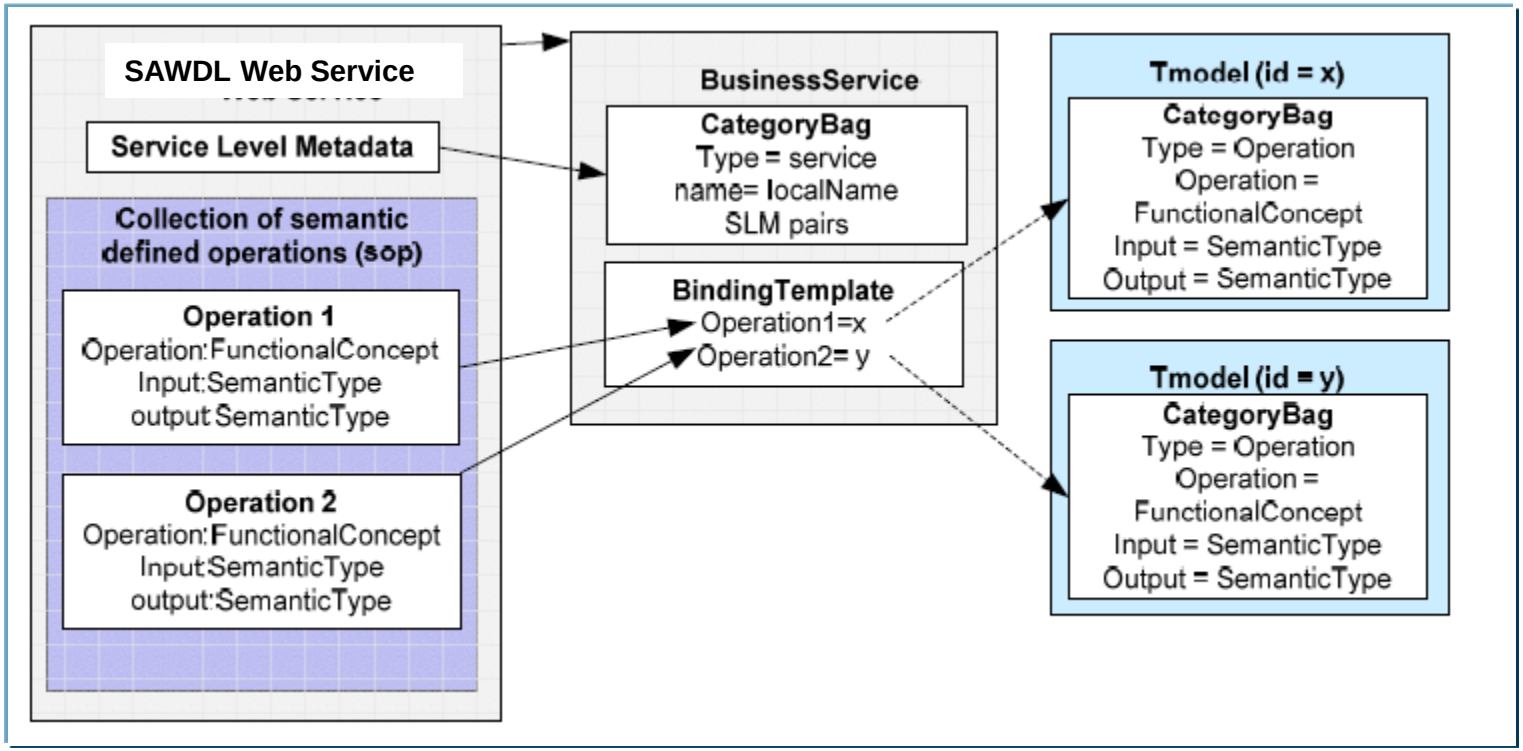
Semantics Templates represented using SAWSDL without implementation details

Basic UDDI Data Structures



Picture taken from: http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v202-20040631.pdf

Mapping SAWSDL to UDDI Data Structures for publication



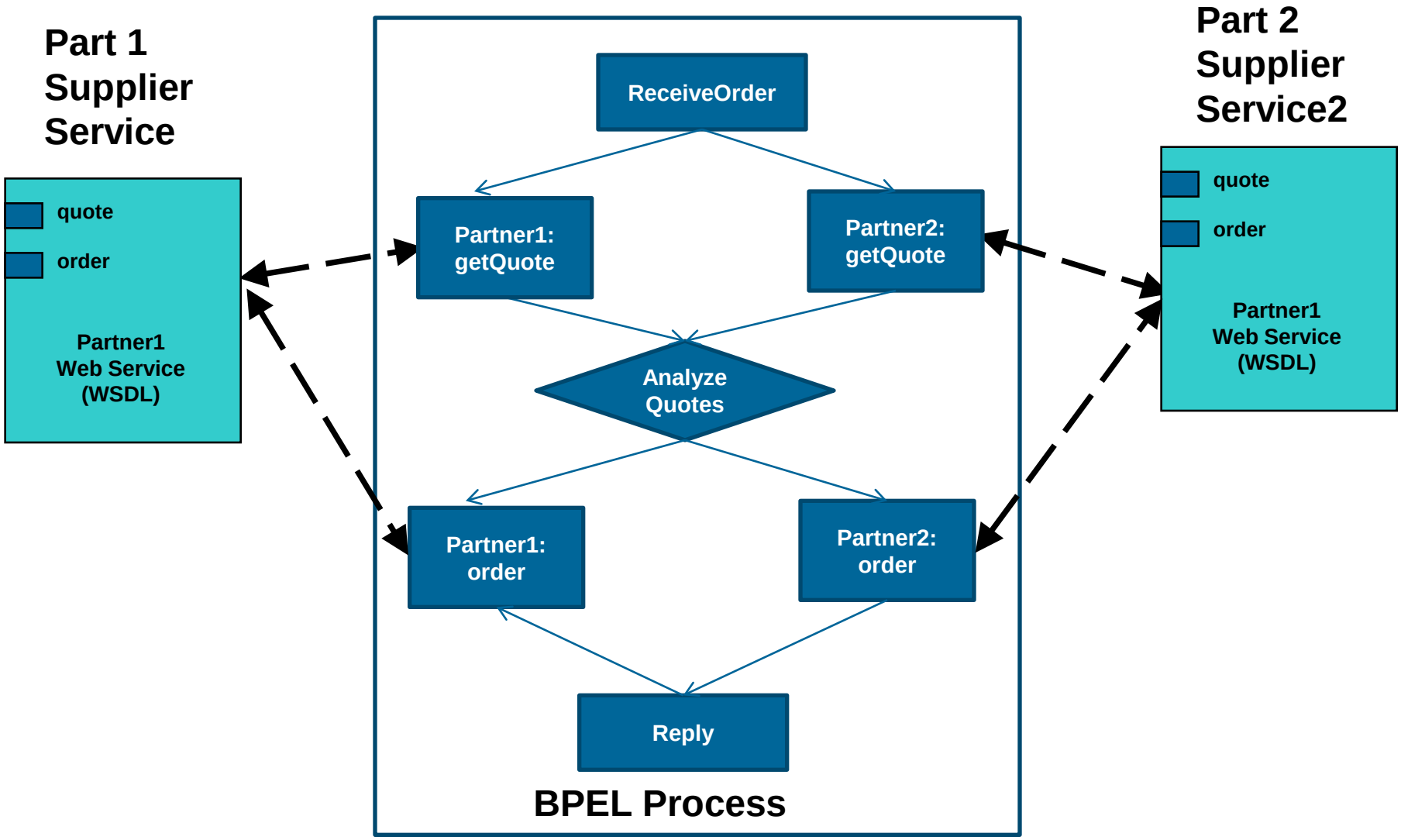
Sample query using UDDI API

```
<find_tModel generic="2.0" xmlns="urn:uddi-org:api_v2">
  <categoryBag>
   <keyedReference tModelKey="WSDL_TYPE_T_MODEL_KEY"
       keyName="WSDL type"
       keyValue="operation"/>
   <keyedReference tModelKey="OPERATION_MODELREF_TMODEL_KEY"
      keyName="Operation-modelReference"
      keyValue= "http://example.org/rosetta#RequestPurchaseOrder"/>
  <keyedReference tModelKey="INPUT_MODELREF_TMODEL_KEY"
     keyName="Input-modelReference"
     keyValue= "http://example.org/rosetta#PurchaseOrderRequest"/>
  <keyedReference tModelKey="OUTPUT_TMODEL_KEY"
    keyName="Output-modelReference"
    keyValue= "http://example.org/rosetta#PurchaseOrderConfirmation"/>
 </categoryBag>
</find_tModel>
```

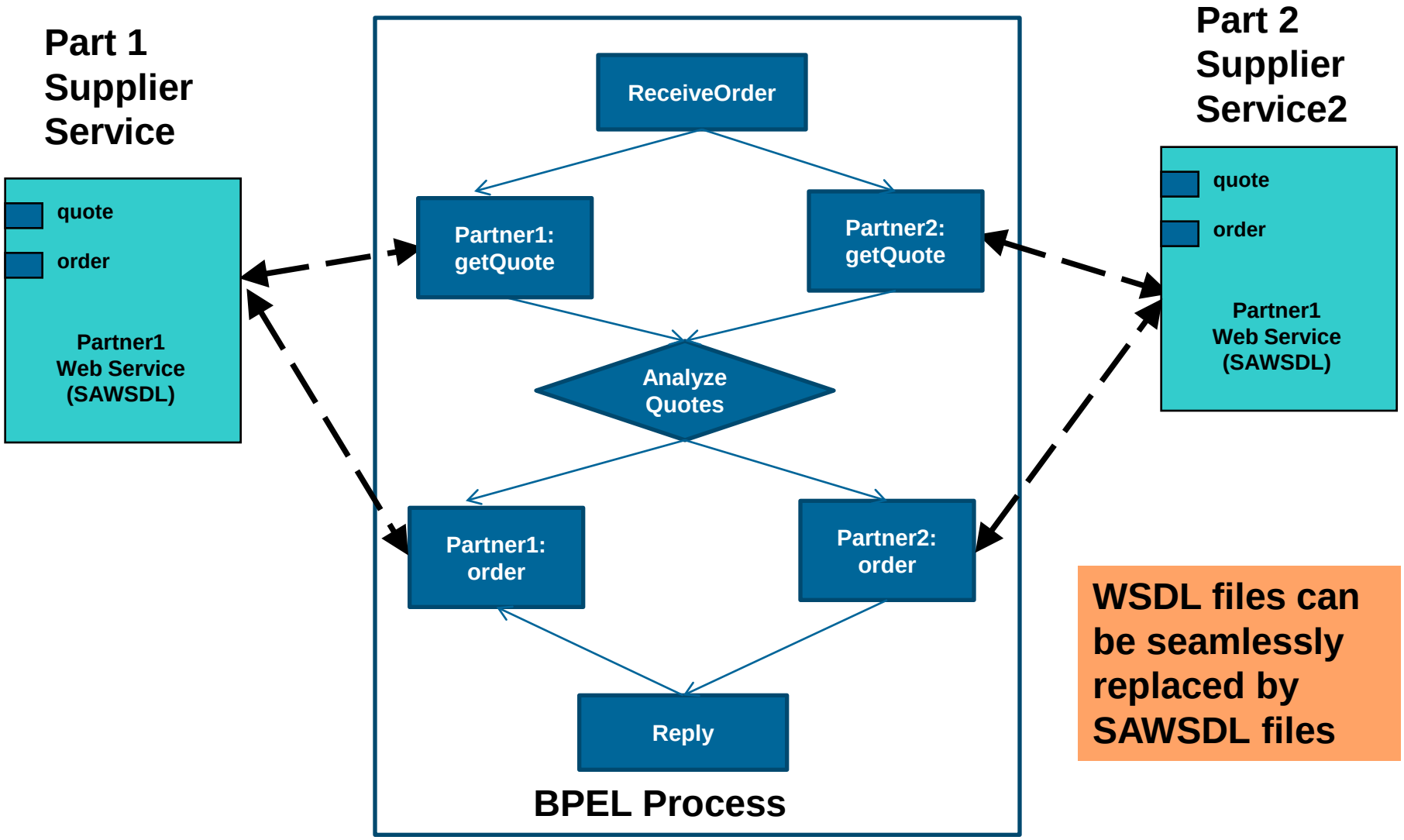
Part 3.2

USING SAWSDL WITH WS-BPEL FOR DYNAMIC BINDING

Sample BPEL Process with WSDL services



Sample BPEL Process with SAWSDL services

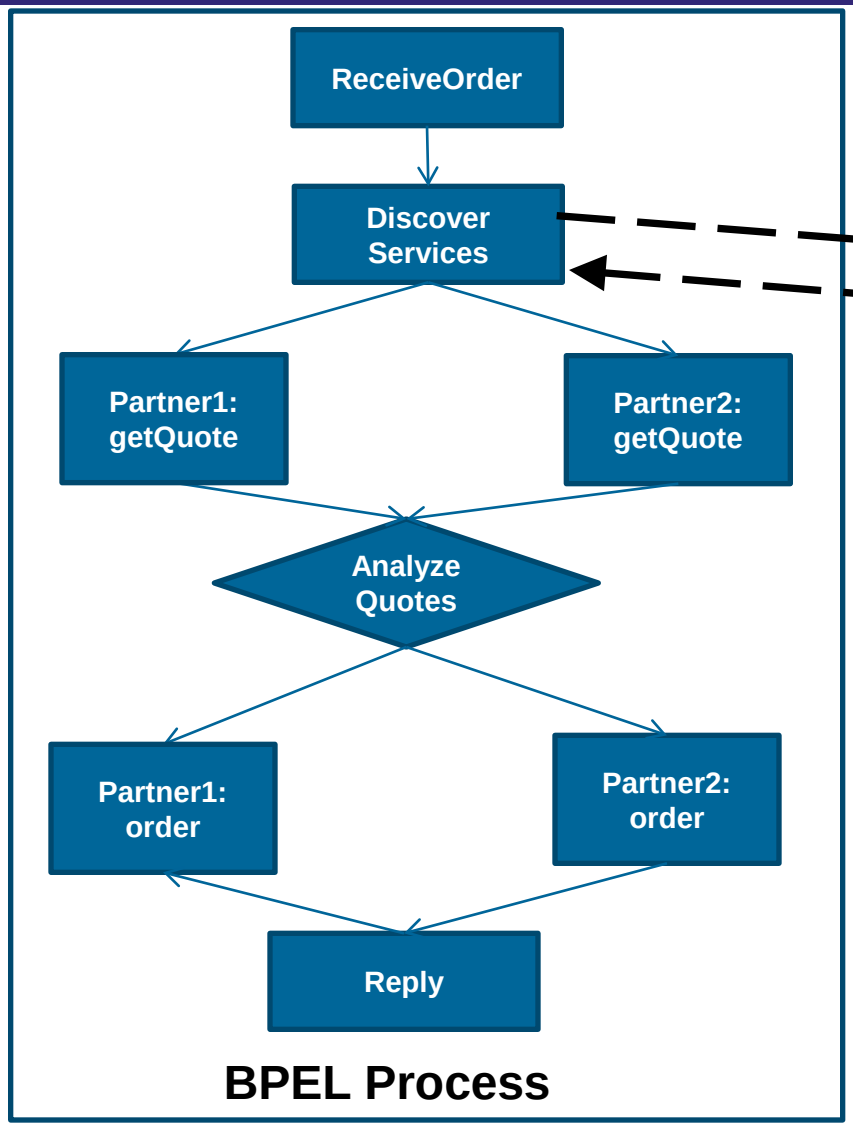


Advantages of using SAWSDL with WS-BPEL

- Can link actual SAWSDL based Web services or semantic templates.
- SAWSDL based Web services can be used to replace existing services in case of business or physical failures
- SAWSDL based semantic templates can be used for dynamic binding:
  - They can be replaced with actual services either at run-time or deployment time to achieve dynamic binding
- BPEL already has support for dynamic binding. SAWSDL provides a mechanism to represent functional semantics of a service, that can be used for discovery.

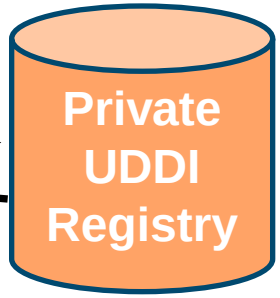
Kunal Verma, Configuration and Adaptation of Semantic Web Processes, Ph.D. Thesis, Department of Computer Science, The University of Georgia, August 2006

Sample BPEL Process with SAWSDL services



Process sends service discovery criteria based on SAWSDL of partner services

Registry responds with list of service matching criteria



Advantages:

- **Services can be added or deleted by publishing them to the registry. Process doesn't have to be changed and re-deployed**
- **Process can be re-used for different types of orders by changing service discovery criteria**

accenture

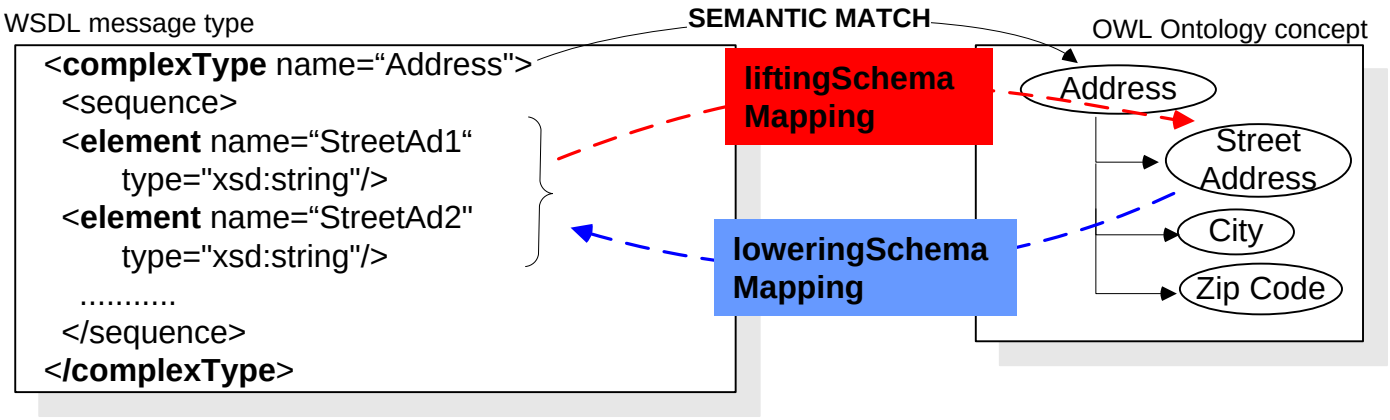
kno.e.sis

Part 3.3

USING SAWSDL FOR DATA MEDIATION

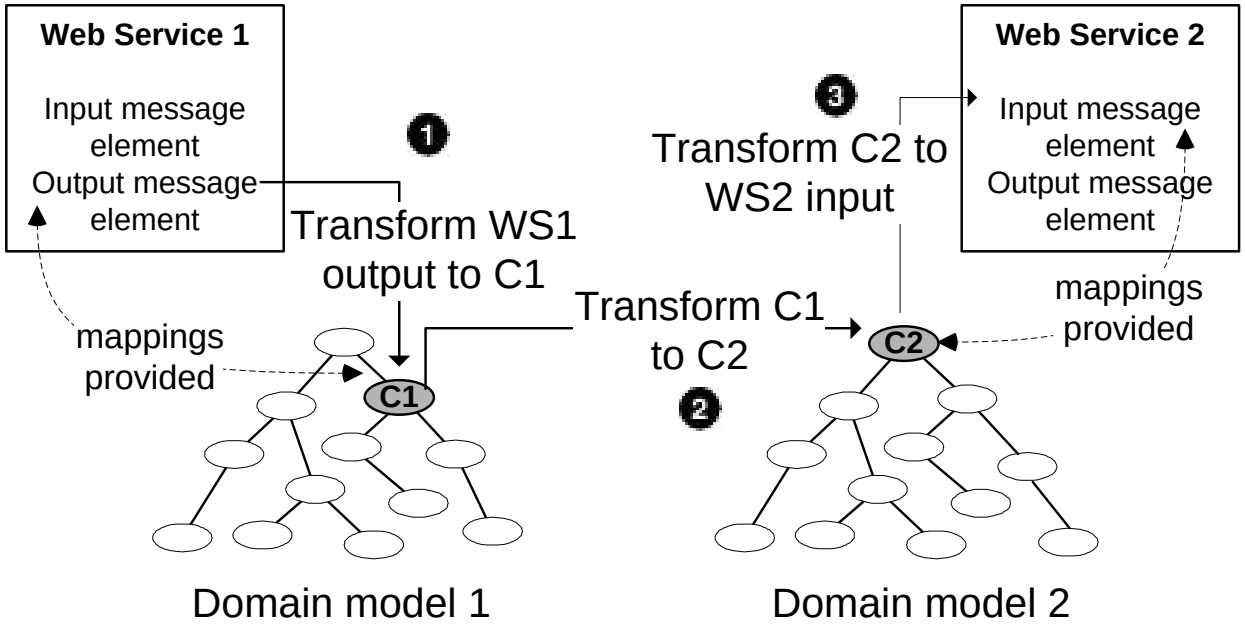
SAWSDL support for data mediation

- User specified mappings from Web service message element to semantic model concept (say OWL Ontology)
  - *liftingSchemaMapping:* from WS message element to OWL concept
  - *loweringSchemaMapping:* from OWL concept to WS message element



Realizing data mediation

- Web services interoperate by re-using these mappings.
  - Ontologies now a vehicle for Web services to resolve message level heterogeneities



Prototype implemented using METEOR-S

- METEOR-S Middleware
  - EPR handler – End Point Resolution handler
    - For clients to use the middleware
    - Reroute SOAP messages to middleware
  - DM handler – Data Mediation handler
    - Main component for facilitating data mediation
    - Works with the EPR handler + a mapping processing engine (SAXON for XQuery / XSLT)

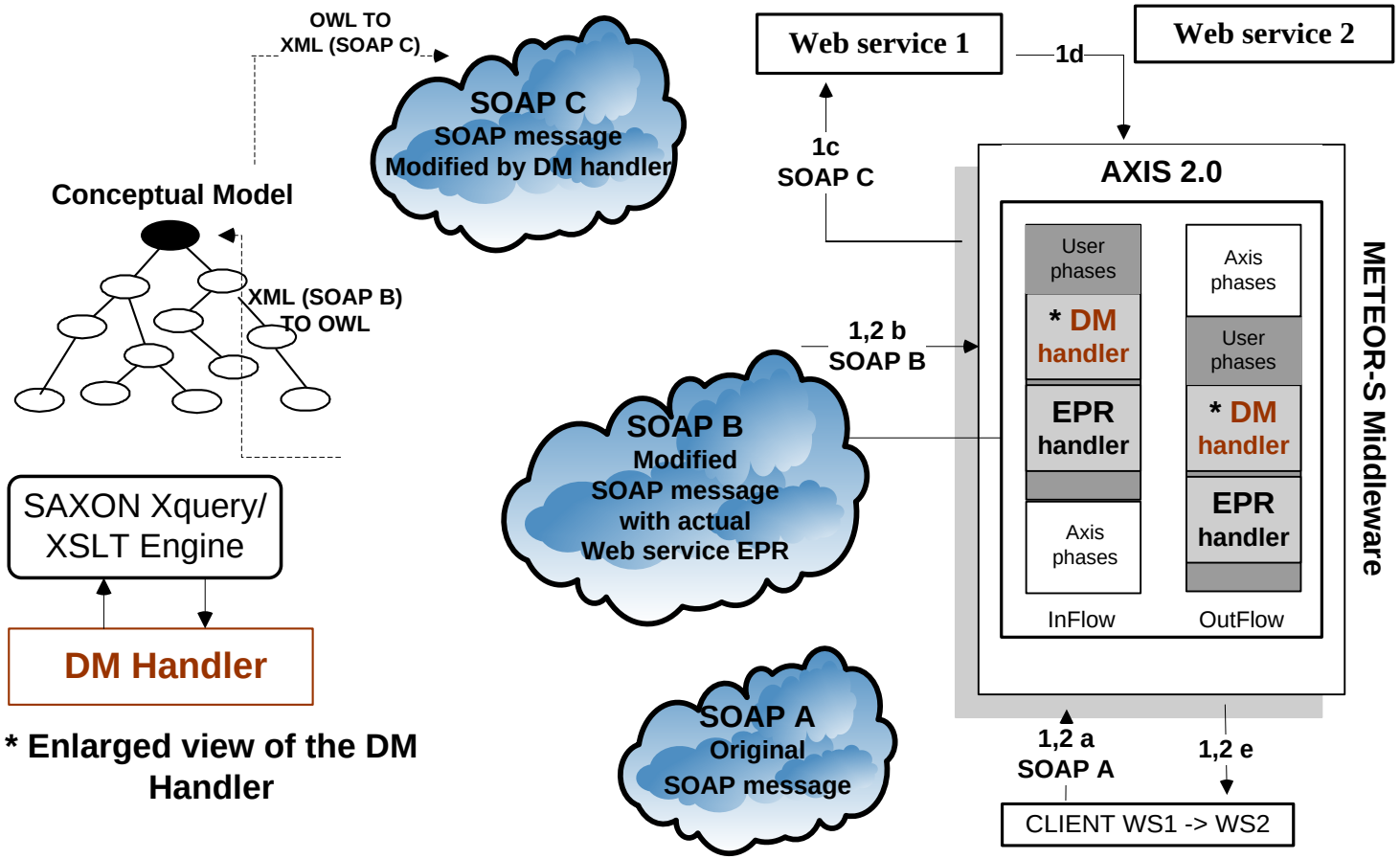
Meenakshi Nagarajan, Kunal Verma, Amit P. Sheth, John A. Miller, Jon Lathem: Semantic Interoperability of Web Services - Challenges and Experiences. ICWS 2006: 373-382

DM Handler – a closer look

- Each time a Web service is invoked
  - obtains the *'liftingschemaMapping*' and *loweringschemaMapping*' functions from SAWSDL locations (using the SAWSDL4J API)
  - performs the lifting and lowering transformations on the incoming SOAP message using a mapping processor/engine (SAXON for XQuery and XSLT)
  - updates the SOAP message. Appropriate Axis handlers then invoke the Web service with the transformed message.

Note: This implementation used WSDL-S, so it used upcast and downcast instead of *liftingschemaMapping*' and *loweringschemaMapping*'
SAWSDL4J API available at: http://knoesis.wright.edu/opensource/sawsdl4j/

Walk through example – WS1 invocation



Part 4

SAWSDL TOOLS

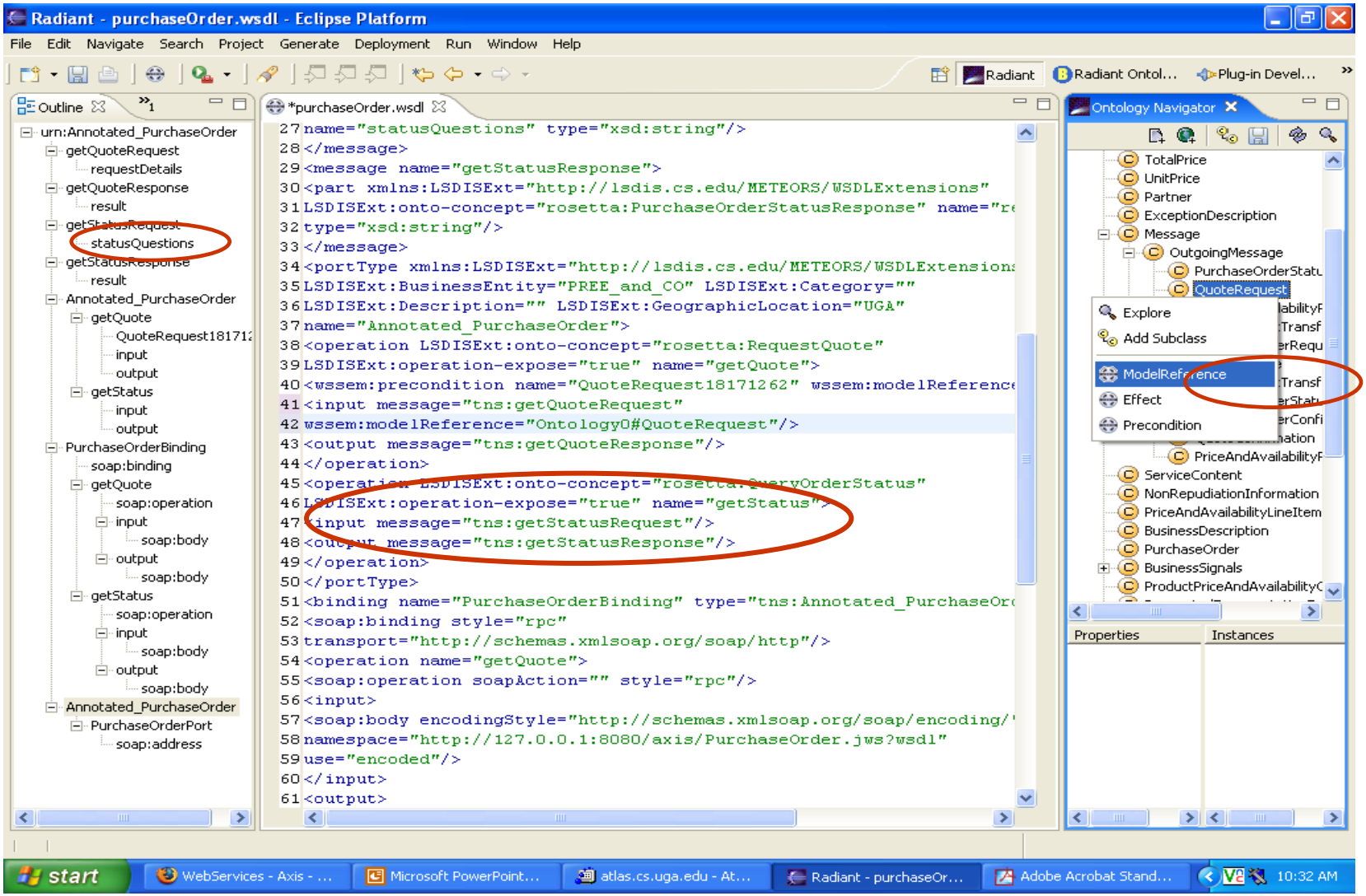
Outline

- Part 1 – Background and Motivation
- Part 2: Deep Dive into SAWSDL
- Part 3: Using SAWSDL
- Part 4: SAWSDL Tools
  - METEOR-S Tools
  - IBM Tools
  - DERI Tools

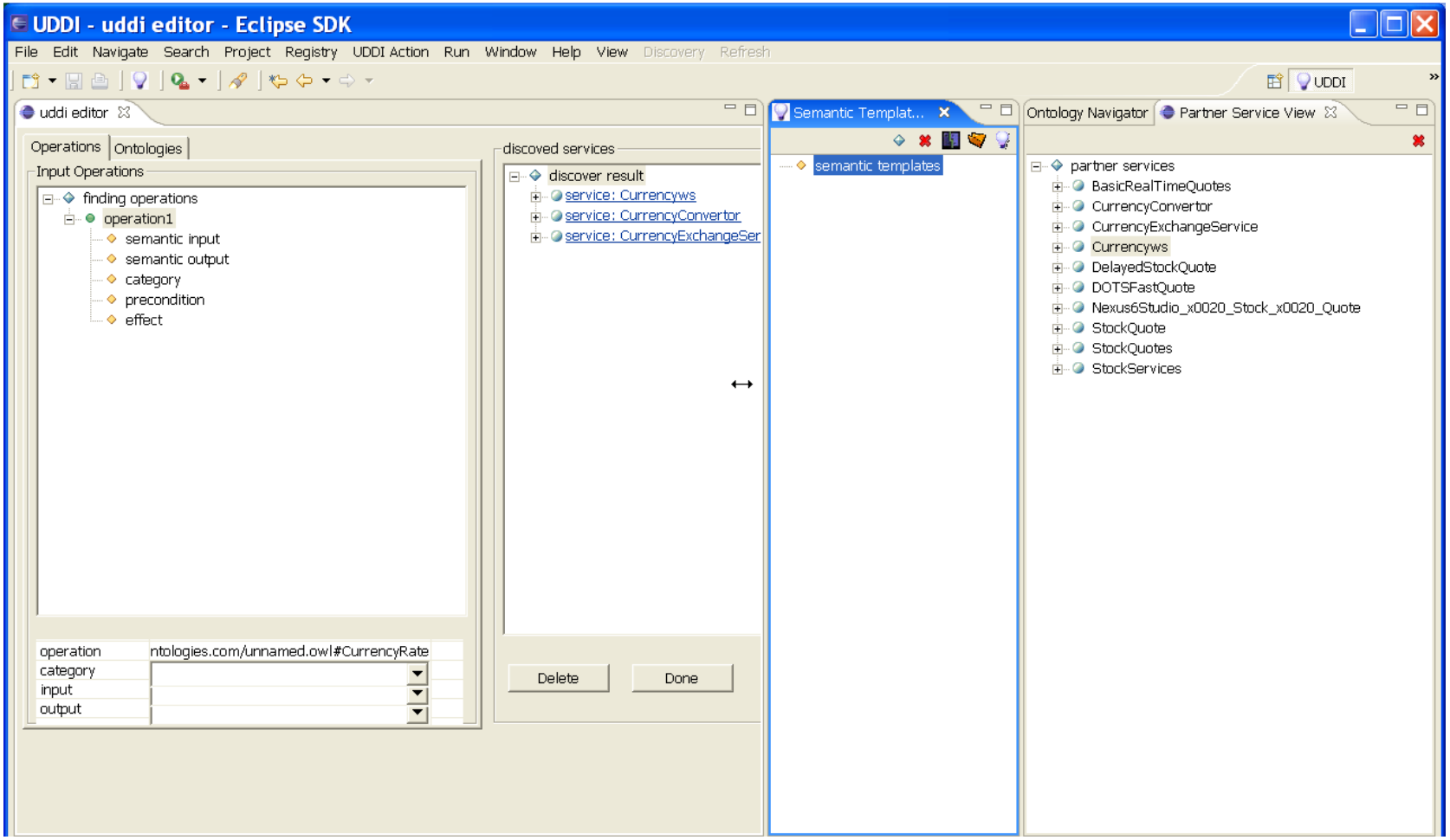
METEOR-S Tools

- A number of tools are available from the METEOR-S project at the University of Georgia and Wright State University
  - SAWSDL4J: API for manipulating SAWSDL files. Nightly builds available for download at http://knoesis.wright.edu/opensource/sawsdl4j
  - Radiant: WSDL-S/SAWSDL Annotation Tool. Version 1.0 available for download at http://lsdis.cs.uga.edu/projects/meteor-s/downloads/index.php?page=1
  - Semantic Services Registry**:** A distributed registry with SOAP and REST bindings for publishing, versioning and discovering SAWSDL services. Nightly builds will be available in Fall 2007.
  - Axis2 Plugins for Dynamic Binding: Plugins to Apache Axis 2 to support dynamic binding of services. Nightly builds will be available in late Fall 2007.
- DEMO/OVERVIEW

Semantic Annotation and Publication - Radiant



Semantic Web Services Discovery: Lumina



IBM Tools

- A number of tools from IBM are available for WSDL-S/SAWSDL at http://www.alphaworks.ibm.com/tech/wssem

- Tools are available as part of the ETTK toolkit for the following features:
  - Web Service Interface Matching
  - Web Service Discovery
  - Web Service Composition

- DEMO/OVERVIEW

DERI Tools

accenture

kno.e.sis

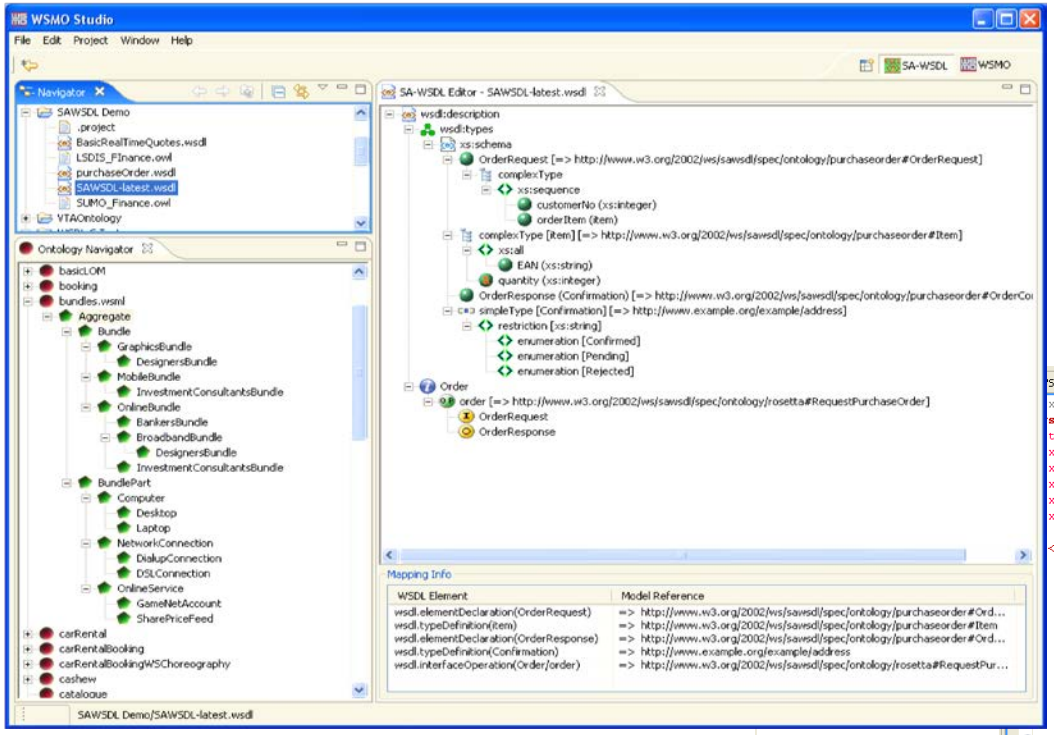
- Integrated with WSMO Studio
  - Support for all modelRef annotations
  - Limited lifting/lowering schema support
    - No support for XSLT / SPARQL mapping definitions
  - Support for WSDL 1.1
- User Interface
  - Tree view of the WSDL structure
  - Text view with syntax highlighting
  - Online demo at http://www.wsmostudio.org/demo/sawsdl.htm

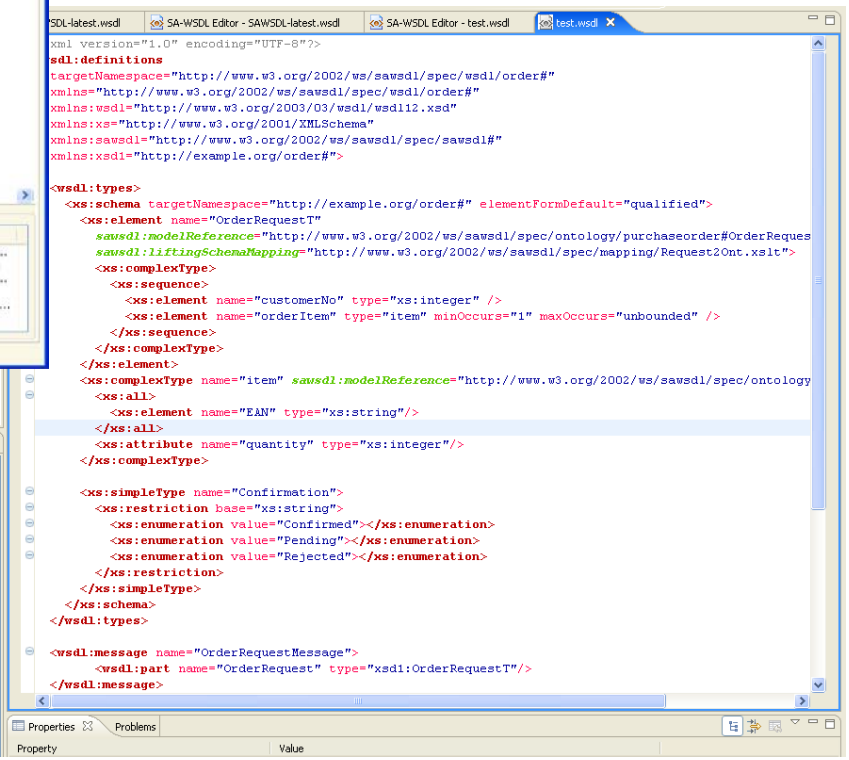
- DEMO/OVERVIEW

WSMO Studio

- SWS Modelling Environment for WSMO
  - Java & Eclipse based
  - Open source (LGPL)
  - http://www.wsmostudio.org
- Components
  - WSMO editor
  - Choreography designer
  - SAWSDL editor
- Features
  - Import/export (WSML / RDF / OWL-DL)
  - Integrated WSML reasoners (Pellet, MINS, KAON2)
  - Front-end to SWS repositories (ORDI, IRS-III, WSMX)
  - Front-end to SWS matchmakers (EPFL)

WSMO Studio / SAWSDL Edtor





Conclusions & Next Steps

- SAWSDL on its way to be a W3C recommendation
- SAWSDL is being evaluated by a number of businesses.
- WSDL-S idea of semantic annotation is leading to several other annotations in SOA: SAREST (being developed at Wright State University & UGA), semantic annotation of policy descriptions, etc.
- Some additional work being taken up by W3C Incubator on SWS-testbed http://www.w3.org/2005/Incubator/swsc/

Acknowledgements

- We would like to thank the following individuals for providing slides for this tutorial:
  - Rama Akkiraju, IBM T.J. Watson Research Center (WSDL-S team and W3C SAWSDL Committee Member)
  - Jacek Kopecky, DERI, (W3C SAWSDL Committee Chair)
  - Marin Dimitrov, DERI
  - John Miller, UGA (WSDL-S team and SAWSDL committee member)
  - Meenakshi Nagarajan, Kno.e.sis. Center, Wright State University (WSDL-S team member)